On Neural Networks as Infinite Tree-Structured Probabilistic Graphical Models

[1a]Boyao Li, M.B.; [2b]Alexander J. Thomson, M.B.; [3c]Houssam Nassif, Ph.D.; [1d]Matthew M. Engelhard, M.D., Ph.D.; [1e]David Page, Ph.D.
[1]Department of Biostatistics and Bioinformatics, Duke University; [2]Department of Computer Science, Duke University; [3]Meta Inc.
{[a]boyao.li, [b]alexander.thomson, [d]m.engelhard, [e]david.page}@duke.edu; [c]houssamn@meta.com

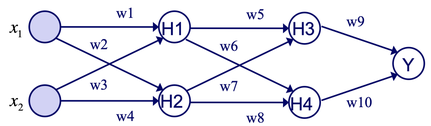
NEURAL INFORMATION PROCESSING SYSTEMS

Introduction

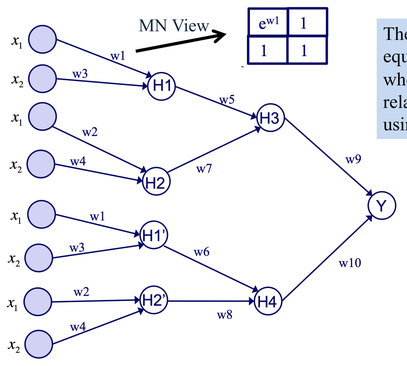
Deep neural networks (DNNs) lack the precise semantics and definitive probabilistic interpretation of probabilistic graphical models (PGMs). We propose an innovative solution by constructing infinite tree-structured PGMs that correspond exactly to neural networks. Our research reveals that DNNs, during forward propagation, indeed perform approximations of PGM inference that are precise in this alternative PGM structure. Not only does our research complement existing studies that describe neural networks as kernel machines or infinite-sized Gaussian processes, but it also elucidates a more direct approximation that DNNs make to exact inference in PGMs. Potential benefits include improved pedagogy and interpretation of DNNs, and algorithms that can merge the strengths of PGMs and DNNs.

Methods

❖ Construction of Tree-Structured PGMs for all-sigmoid DNNs



Step 1: When any node (e.g., H1 or H2) has multiple common children (H3, H4), give each child its own copy of that node (e.g., H1 and H1') and ancestors.



The Bayesian network view of a DNN is equivalent to the Markov network view, where directionality is removed and the relationships between nodes are defined using the potential shown here.

Step 2: Working from input to output, replace each edge A->B with **L copies** of the edge, of A and A's entire ancestor structure. Where A->B has weight w, **all its copies have weight w/L**.

Theorem 1 (Matching Probabilities): In the PGM construction, for an arbitrary latent node h in the DNN that has observed parents $g_1, \dots, g_M$ and latent parents $h_1, \dots, h_N$ that are true with probabilities $p_1, \dots, p_N$,

$$P(h = 1|X) \xrightarrow{L\to\infty} \sigma\left(\sum_{j=1}^{M} w_j g_j + \sum_{k=1}^{N} \theta_k \sigma(p_k)\right)$$

where w_js' and θ_ks' are the weights on edges between these nodes and h. As L goes to infinity, for every node h anywhere in the network, P(h|X) is identical for NN, Bayesian network (BN), and Markov network (MN). The theorem shows that forward propagation in an all-sigmoid DNN can be viewed as the approximation of the inference in a tree-structured PGM.

Theorem 2 (Matching Gradients): In the PGM construction, as $L \to \infty$, the derivative of the marginal log-likelihood, where all hidden nodes have been summed out, with respect to a given weight exactly matches the derivative of the cross-entropy loss in the neural network with respect to the equivalent weight in its structure.

❖ Theoretical Result Suggests HMC Algorithm that can reason over combined DNN and PGM components

The PGM-view suggests alternative training algorithms such as contrastive divergence with MCMC sampling (e.g., Gibbs)[2]. However, Gibbs sampling on NNs suffers from its high cost of computational resources. Here we use HMC in DNN as a fast approximation to Gibbs in this infinite tree-width-1 PGM.
Based on the PGM construction, as L goes to infinity, the average of sampled L copies for each hidden node follows normal distribution asymptotically by the central limit theorem. When L is finite in practice, the following normal distribution is a reasonable approximation to the probability distribution of hidden nodes (h_i is the vector of the ith layer, and $h_0 = X$):

$$h_i \sim N\left(p_i, \frac{p_i(1-p_i)}{L}\right), where\ p_i = \sigma(W_{i-1}h_{i-1} + b_{i-1})$$

This is used to sample the state of latent variables. We can then calculate the $p(Y, H|X, \theta)$:

$$p(Y, H|X, \theta) = p(Y|h_K, \theta_K) \cdot \prod_{i=1}^{K} p(h_i|h_{i-1}, \theta_{i-1})$$

And the loss function for weight updating is:

$$L(\theta) = -\log p(Y, H|X, \theta)$$

Results

Table 1: Calibration performance on synthetic datasets. Experiments are run on each dataset 100 times to avoid randomness. T-tests are used to test whether Gibbs and HMC have smaller MAE than SGD, and highlighted cells means it is statistically significant to support the hypothesis.

Data (Weight)	# Train Epochs	Average Mean Absolute Error ($\times 10^{-3}$) (p-value)				
		DNN	Gibbs	HMC-10	HMC-100	HMC-1000
BN (0.3)	100	6.593	16.09 (1.0000)	5.300 (<0.0001)	6.864 (0.9982)	6.658 (1.0000)
	1000	34.44	36.69 (0.9916)	23.53 (<0.0001)	34.96 (1.0000)	34.55 (1.0000)
BN (1)	100	22.90	20.84 (0.0011)	22.48 (<0.0001)	24.17 (1.0000)	22.95 (0.9928)
	1000	42.59	33.64 (<0.0001)	33.07 (<0.0001)	43.03 (1.0000)	42.63 (0.9995)
BN (3)	100	72.76	76.12 (1.0000)	76.54 (1.0000)	72.62 (<0.0001)	72.63 (<0.0001)
	1000	28.28	32.59 (1.0000)	32.98 (1.0000)	28.84 (1.0000)	28.40 (1.0000)
BN (10)	100	186.0	192.8 (1.0000)	196.1 (1.0000)	184.6 (<0.0001)	184.8 (<0.0001)
	1000	54.89	79.69 (1.0000)	72.64 (1.0000)	54.81 (0.1266)	54.63 (<0.0001)
MN (0.3)	100	6.031	14.03 (1.0000)	4.515 (<0.0001)	6.382 (1.0000)	6.070 (1.0000)
	1000	38.11	34.83 (<0.0001)	26.54 (<0.0001)	38.71 (1.0000)	38.22 (1.0000)
MN (1)	100	9.671	17.81 (1.0000)	8.887 (<0.0001)	9.018 (<0.0001)	9.284 (<0.0001)
	1000	27.80	32.44 (1.0000)	19.92 (<0.0001)	27.98 (0.9994)	27.73 (<0.0001)
MN (3)	100	8.677	23.60 (1.0000)	5.685 (<0.0001)	5.912 (<0.0001)	5.964 (<0.0001)
	1000	5.413	28.03 (1.0000)	5.792 (0.9957)	5.671 (1.0000)	5.443 (1.0000)

Synthetic datasets are generated by simple BNs and MNs with their weights in different ranges, which are used to define the conditional probabilistic distributions for BNs and potentials for MNs. Each dataset contains 1000 data points which are binary. The true distribution of the corresponding BN/MN is calculated by sampling or the variable elimination algorithm. For different methods, we compare their mean absolute error and apply T-tests to check the statistical significance.

Conclusion

- We have established a new connection between DNNs and PGMs by constructing an infinite tree-structured PGM corresponding to any given DNN, then showing that inference in this PGM corresponds exactly to forward propagation in the DNN.
- We anticipate it will inspire new algorithms that merge strengths of PGMs and DNNs. We have explored one such algorithm, a novel HMC-based algorithm for DNN training motivated by our PGM construction and illustrated how it can be used to improve to improve DNN calibration.
- IGVF needs reasoning with merged DNN and probabilistic graphical model (PGM) components in a unified regulatory model such as that shown below.

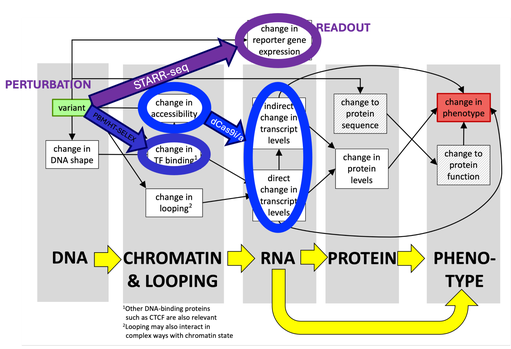


Image Credit: William H. Majoros, Ph.D., Duke Biostatistics and Bioinformatics

References

1. Buhrmester, V., Münch, D., & Arens, M. (2021). Analysis of explainers of black box deep neural networks for computer vision: A survey. Machine Learning and Knowledge Extraction, 3(4), 966-989.
2. Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. Neural computation, 18(7), 1527-1554

Acknowledgements

This project is in part supported by Impact of Genomic Variation on Function (IGVF) Consortium of the National Institutes of Health via grant U01HG011967.